Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in this application.

Listing of Claims:

1-178. (Canceled).

179. (Currently Amended) A method for programming a system having a configurable cellular structure, comprising:

extracting a control flow graph of a program;

separating the control flow graph into a plurality of subgraphs; [[and]]

distributing the plurality of subgraphs among a plurality of programmable hardware modules;

determining state information for each of the subgraphs; and

transferring the state information determined for one of the subgraphs from the one of the subgraphs to a subsequently executed subgraph.

180. (Currently Amended) A method for programming a system having a configurable cellular structure, comprising:

extracting a data flow graph of a program that includes a loop;

separating the data flow graph into a plurality of subgraphs, such that the loop is split into several of the subgraphs; and

distributing the plurality of subgraphs among a plurality of hardware modules, such that ~~;~~ ~~wherein~~ the several ~~separating includes separating the data flow graph so as to minimize connections between the plurality of~~ subgraphs of the loop are distributed among at least two of the hardware modules.

181. (Currently Amended) A method for programming a system having a cellular structure, comprising:

extracting from a program at least one of a data flow graph and a control flow graph;

separating the at least one of the graphs into a plurality of subgraphs; and

distributing the plurality of subgraphs among a plurality of hardware modules;

wherein the separating includes providing communication arrangements adapted for storage ~~separating the at least one~~ of all data to be processed in a subsequent hardware module according to connections between ~~the graphs so as to minimize connections between~~ the plurality of subgraphs.

182. (Canceled).

183. (Previously Presented) The method of claim 181, wherein the separating includes separating the at least one the graphs into the plurality of subgraphs so that data transmission between the plurality of subgraphs is minimized.

184. (Previously Presented) The method of claim 181, wherein the separating includes separating the at least one of the graphs into the plurality of subgraphs so that no loop-back is obtained between the plurality of subgraphs.

185. (Previously Presented) The method of claim 181, wherein the separating includes separating the at least one of the graphs into the plurality of subgraphs so that the subgraphs match resources of the hardware modules.

186. (Currently Amended) The method of claim 181, wherein ~~further comprising: inserting~~ memory elements are inserted between the plurality of subgraphs, the memory elements adapted to save data passed between subgraphs.

187. (Previously Presented) The method of claim 181, wherein each of the plurality of subgraphs includes nodes, the method further comprising:

transmitting status signals between nodes within one of the subgraphs so that a state of each individual one of the nodes of the one of the subgraphs is available to each of the other nodes of the one of the subgraphs.

188. (Previously Presented) The method of claim 181, wherein each of the plurality of subgraphs includes nodes, the method further comprising:

transmitting status signals from a first node of at least one of the plurality of subgraphs to a higher-level unit adapted to control configuration of the plurality of hardware modules so as to trigger reconfiguration.

189. (Previously Presented) The method of claim 181, wherein the extracting includes, for a conditional instruction, extracting a plurality of different subgraphs, each representing a different instruction path, one of the different subgraphs being executed depending on an evaluation of the conditional instruction.

190. (Currently Amended) A method of executing a single program on a system having an array of runtime reconfigurable cells, comprising:

separating the single program into several subgraphs;

distributing the several subgraphs among different cells of the array; and

executing the several subgraphs via the cells, the executing including:

transmitting a data signal from a first cell via which a first one of the subgraphs is executed to a second cell via which a second one of the subgraphs is executed; and

transmitting a status with the data signal, the status indicating whether the data signal is valid.

191. (Previously Presented) The method of claim 190, further comprising:

receiving a valid data signal at the second cell; and

acknowledging receipt of the valid data signal.

192. (Previously Presented) The method of claim 191, further comprising,

transmitting by the second cell an indication that a signal is expected.

193. (Previously Presented) The method of claim 192, further comprising:

transmitting by the first cell an indication that the first cell is transmitting the expected signal.

194. (Currently Amended) A method of executing a program on a runtime reconfigurable array of cells, the method comprising:

forming a plurality of subgraphs based on a program;

computing a first part of a first one of the subgraphs with a first cell;

after the computing, reconfiguring the first cell for computation of a first part of a second one of the subgraphs; and

simultaneously with the reconfiguring, computing a second part of the first subgraph with a second cell;

wherein state information determined for one of the subgraphs is transferred from the one of the subgraphs to a subsequently executed subgraph.

195. (Previously Presented) The method of claim 194, further comprising:

storing configurations for the first one of the subgraphs and the second one of the subgraphs in configuration registers associated with the first cell.

196. (Previously Presented) The method of claim 195, further comprising:

marking unconfigured ones of the configuration registers as unconfigured.

197. (Previously Presented) The method of claim 194, further comprising:

selecting a configuration for the first cell based on a status signal generated by the cell structure.

198. (Previously Presented) The method of claim 194, further comprising:

selecting a configuration for the first cell based on a status signal generated by a higher-level loading unit.

199. (Previously Presented) The method of claim 194, further comprising:

selecting a configuration for the first cell based on an externally generated status signal.

200. (Previously Presented) The method of claim 194, further comprising:

selecting a configuration for the first cell as a function of a present configuration of the first cell and a received status signal.

201. (Previously Presented) The method of claim 194, further comprising:

activating an unconfigured configuration register in the first cell;

requesting a configuration from a higher-level load unit when the unconfigured configuration register is activated; and

suspending execution of a subgraph until the requested configuration is fully loaded.

202. (Previously Presented) The method of claim 194, further comprising:

triggering a loading of a configuration of the first cell when a status signal generated by the cell structure is received by the first cell.

203. (Currently Amended) A method for programming a system having a runtime configurable cellular structure, comprising:

extracting from a program at least one of a data flow graph and a control flow graph;

separating the at least one of the graphs into a plurality of subgraphs; and

distributing the plurality of subgraphs among a plurality of hardware modules;

wherein:

the extracting includes, for a conditional instruction of the program, extracting a plurality of different subgraphs, each representing a different instruction path of the conditional instruction, the conditional instruction indicating which of the executed instruction paths is to be selected for providing output of the selected instruction path output to a further subgraph; [[and]]

for each one of the different subgraphs, the system sets execution of the subgraph to be bypassed as soon as ~~a~~n evaluation in accordance with the conditional instruction reveals that output of the subgraph will not be selected; and

the distribution of the plurality of subgraphs includes adapting the plurality of hardware modules such that state information determined for a first one of the subgraphs is transferred from the first one of the subgraphs to another subgraph that is to be subsequently executed.